Sandia National Laboratories

Exceptional service in the national interest

ME469: High Performance Computing for CFD

Stefan P. Domino[1,2]

[1] Computational Thermal and Fluid Mechanics, Sandia National Laboratories

[2] Institute for Computational and Mathematical Engineering, Stanford

SAND2018-4536 PE

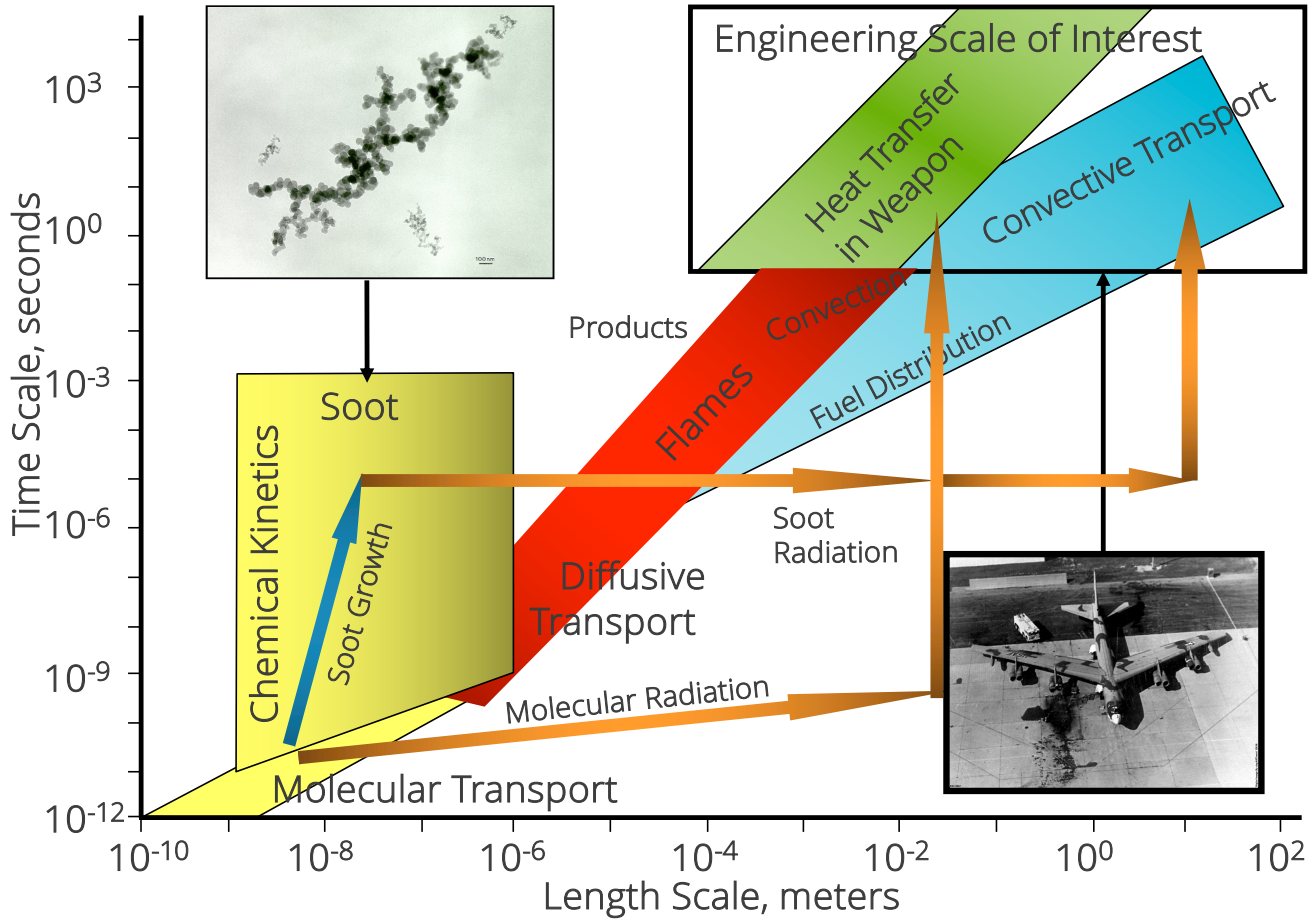
U.S. DEPARTMENT OF ENERGY

NNSA

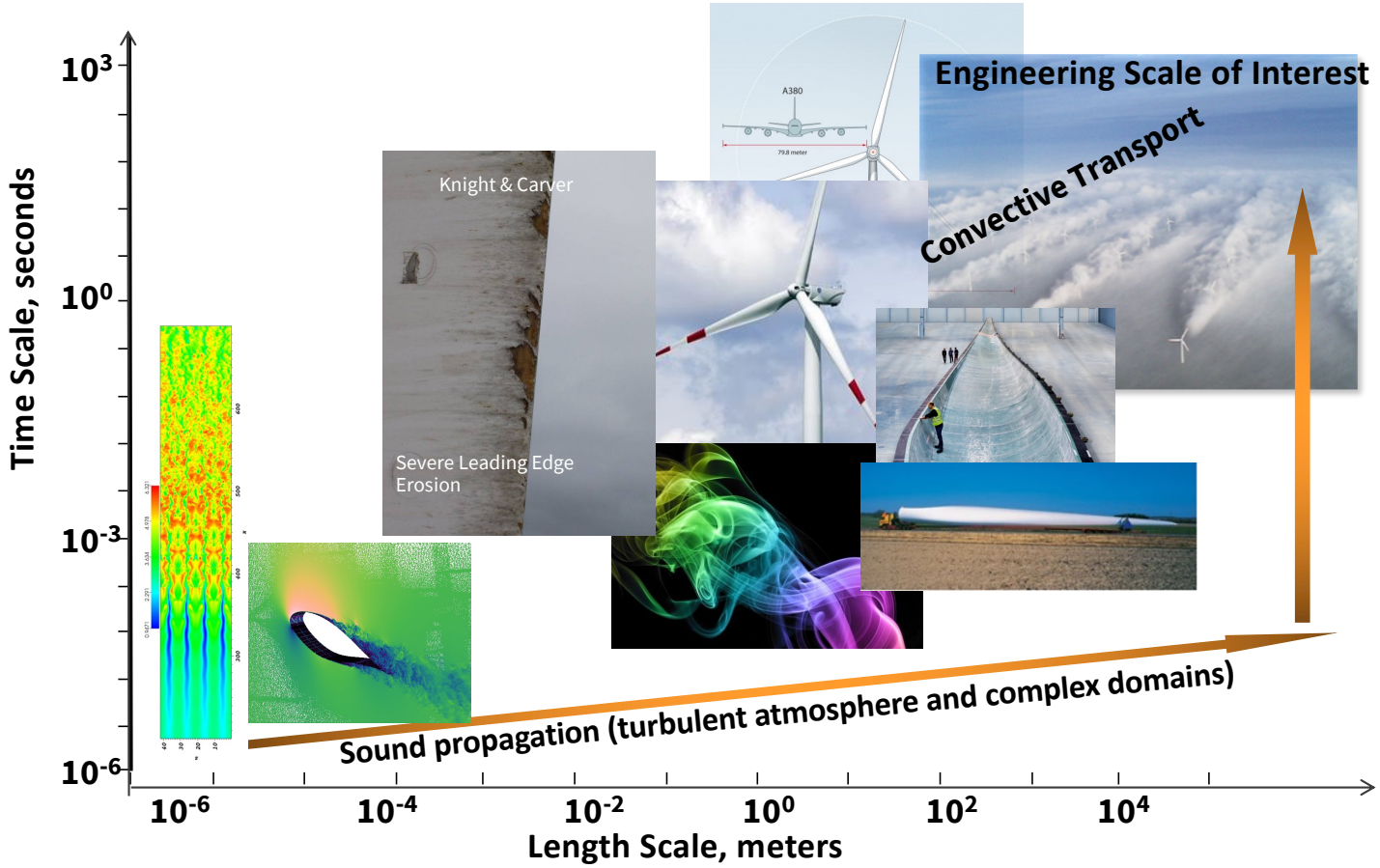
High Performance Computing for CFD: Outline

- Overview of Time and Length Scales
  - Fire
  - Wind
- Conceptual Parallel Computing Model
- Types of Scaling: Strong and Weak
- Examples of Scaling
- Next Generation Platform Exascale Drivers
- Data Movement vs Data Execution
- Conclusions

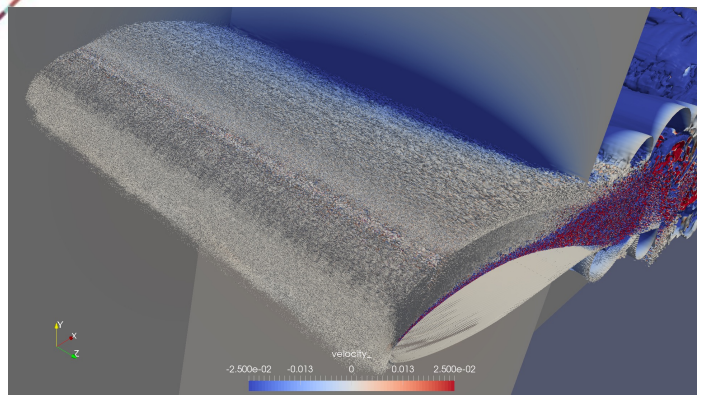
Disparity in Time and Length Scales, Fire

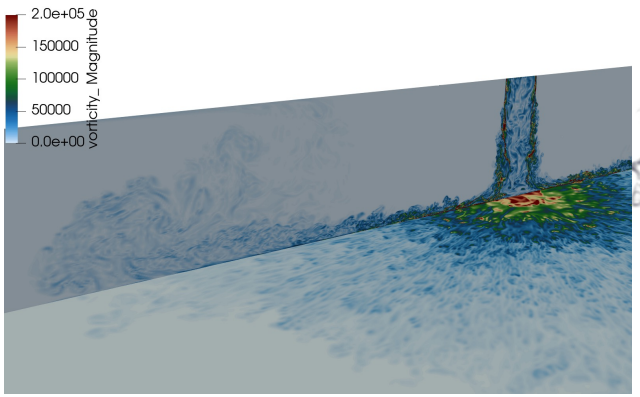


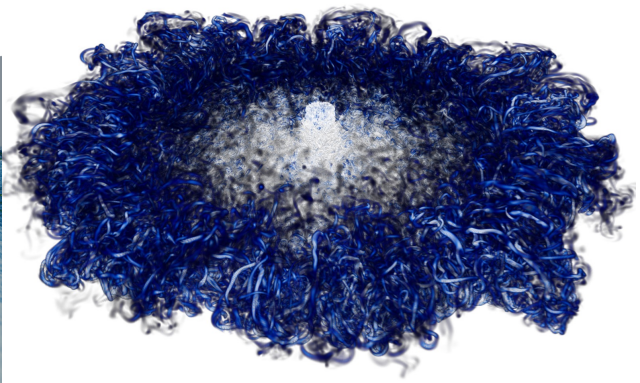
Disparity in Time and Length Scales, Wind

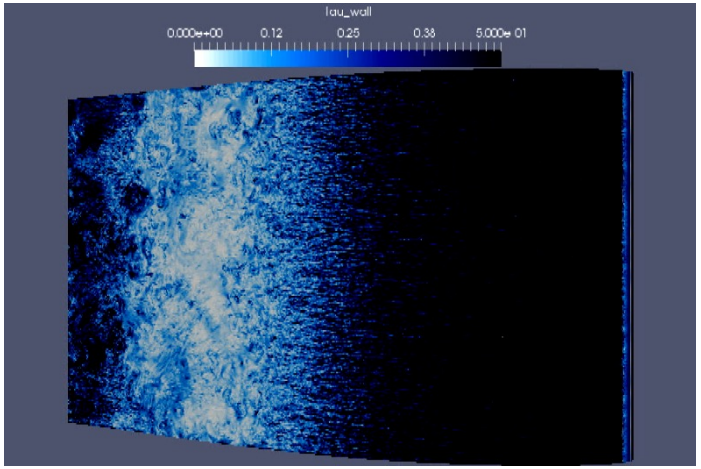


Routine Examples of HPC



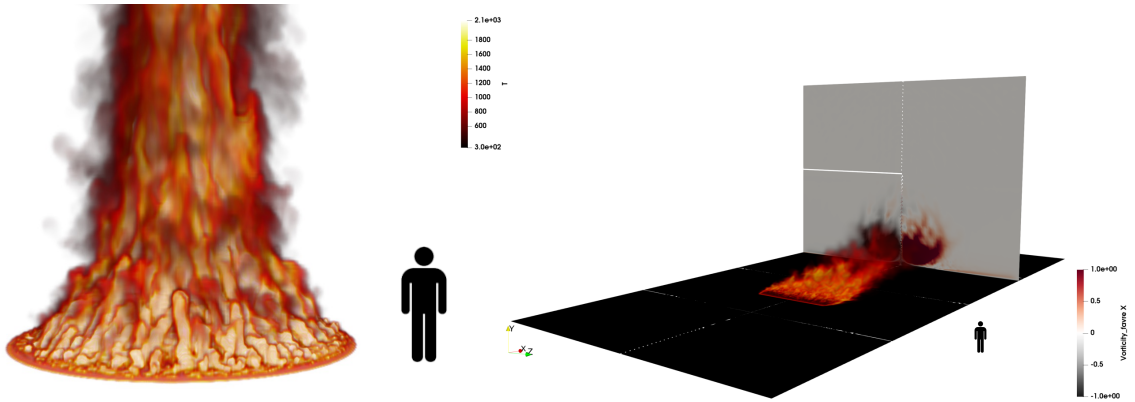






O(6) billion wind energy application

O(2) billion DNS impinging jet, velocity (vertical) and heat flux; Q-criterion



O(200) million multi-physics fire

Conceptual Scaling Exercise

= 500 sq ft/day

= 500 sq ft/day

Painting Rate Legend

- Goal: Paint a house in one day
  - My House: 1000 sq ft
- WH: 60,000 sq ft

- One person would paint my house in two days (!!)
- The white house will require one person 120 days (!!)

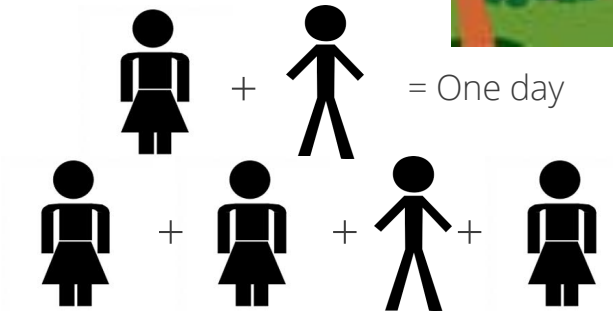
Single House, More People (each person with reduced load)

- Goal: Paint a house in one day
  - My House: 1000 sq ft
- WH: 60,000 sq ft

= 500 sq ft/day

= 500 sq ft/day

Painting Rate Legend

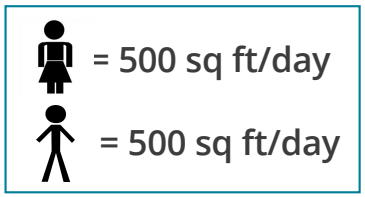


= One day

= Faster Completion; Half day (each person 250 sq ft)

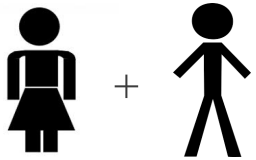
White House, More People (with shared load)

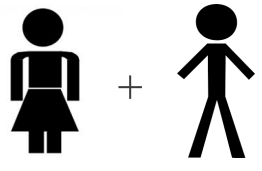
- Goal: Paint a house in one day
  - My House: 1000 sq ft
- WH: 60,000 sq ft



Painting Rate Legend

Time to completion is one day. Everyone involved provided 500 sq ft


= 60 days

60x  = 1 day

Single House, More People (with impediments)

= 500 sq ft/day

= 500 sq ft/day

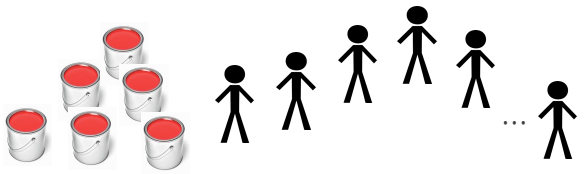
Painting Rate Legend

- Sam's team, who played basketball last night, have some injuries

~ 250 sq ft/day

- Limited paint brushes…

- Plenty of paint, poor loading capacity



- No coherent plan of attack

+ = 60 days 60x (+ + n) >> 1 day

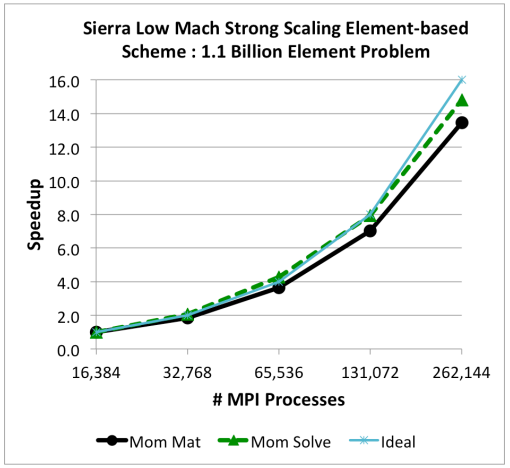
Types of Scaling

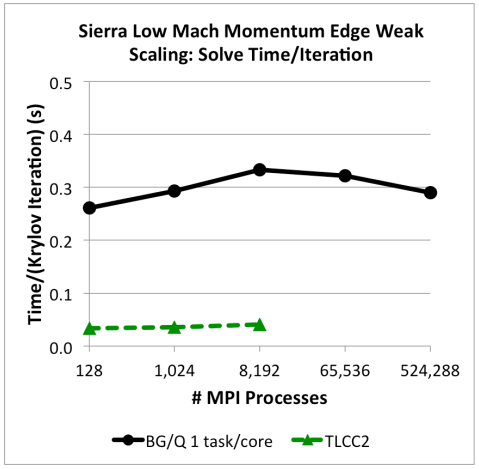
Strong Scaling: How the solution time varies with increased computational resources (cores, threads, GPU/Warps) on a fixed-size problem

- User Y has a mesh that is 1 billion elements and would like to minimize the time it takes to complete a simulation; the larger the resource, the faster the turn-around

Weak Scaling: How the solution time varies with increased problem size on a fixed computational resource load (cores, threads, GPU/Warps)

- User Y is conducting a validation study that includes three mesh resolutions, which were obtained by uniform mesh refinement, and would like to increase the computational resource appropriately for each subsequently refined mesh

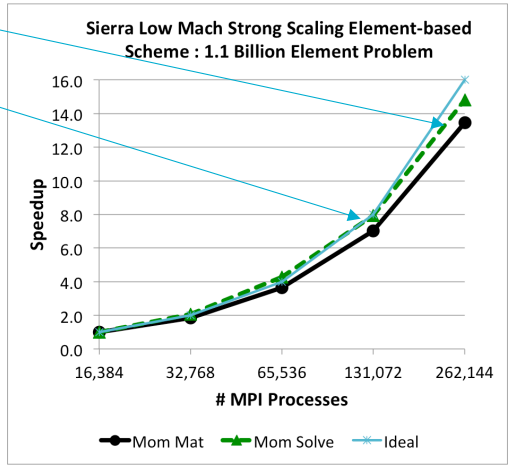




Types of Scaling

~4,000 elem/core

~8,000 elem/core

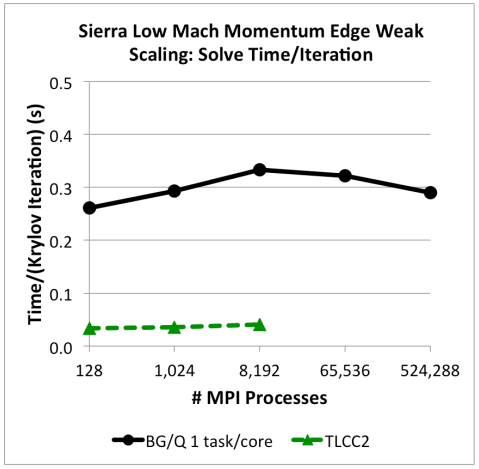


Strong Scaling: How the solution time varies with increased computational resources (cores, threads, GPU/Warps) on a fixed-size problem

- User Y has a mesh that is 1 billion elements and would like to minimize the time it takes to complete a simulation; the larger the resource, the faster the turn-around

Weak Scaling: How the solution time varies with increased problem size on a fixed computational resource load (cores, threads, GPU/Warps)

- User Y is conducting a validation study that includes three mesh resolutions, which were obtained by uniform mesh refinement, and would like to increase the computational resource appropriately for each subsequently refined mesh

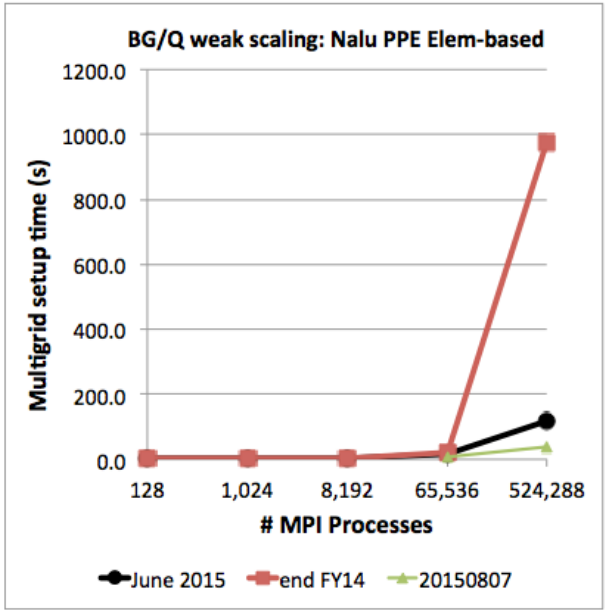


General Implicit Guidelines:
1e6 elem/node
~25k elem/core

Challenges Associated with Scaling

- Consider a communication-intensive code procedure: Algebraic Multigrid (AMG) preconditioner setup
  - Hierarchy of parallel
- Point-to-point communications

Like verification, the product of a first-time scaling study at a new production scale is generally met with work!



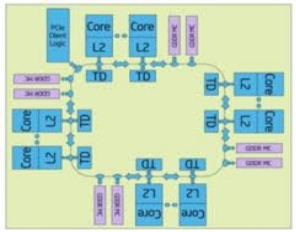
Computing Performance: Through the Years

HPC is facing a new disruption in technology

Mainframes
60's to 70's

Vector
Supercomputers
70's to early 90's

Massively parallel processor (MPP) systems with simple nodes
1990's to 2010



MPP w/ Advanced Architecture nodes: Multilevel, heterogeneous, energy and memory constrained

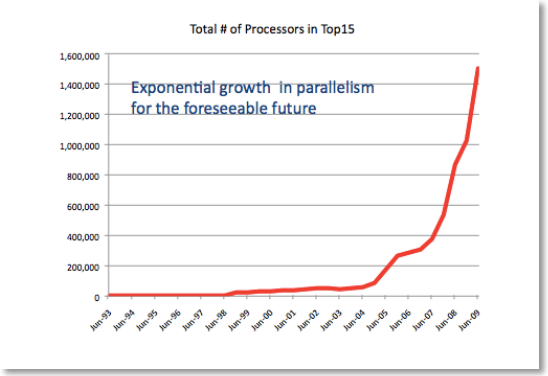
ROOM → MACHINE → CABINET → CHIP

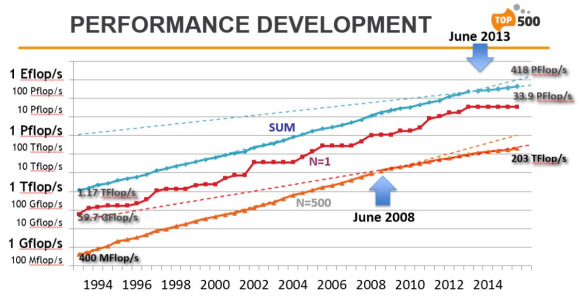
COMPLEXITY

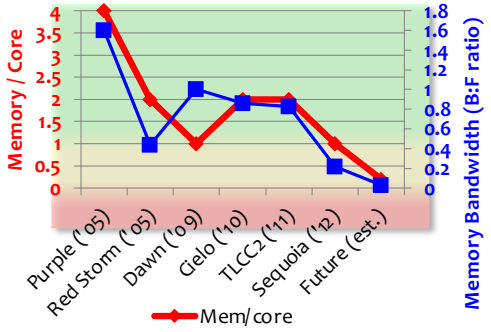
Technology disruptions require a significant increase in complexity of our codes

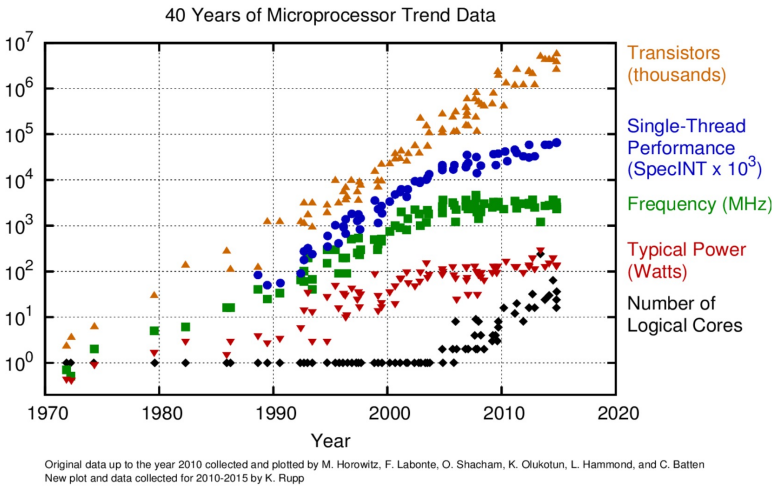
Slides courtesy of Mike Glass, Jim Ang, Rob Hoekstra, Si Hammond, Ron Brightwell, and Stephen Olivier.

Drive Towards Next Generation Platforms









Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten
New plot and data collected for 2010-2015 by K. Rupp

Dramatic increase in on-node parallelism and reduction in relative data movement is counter to our current code performance

Power constraints are driving the rapid increase in on-node parallelism. However, per core memory bandwidth is decreasing making it extremely difficult to fully utilize additional cores

Slides courtesy of Mike Glass, Jim Ang, Rob Hoekstra, Si Hammond, Ron Brightwell, and Stephen Olivier.

High Performance Computing (HPC): Enabling Scientific Advancement

Late Roman Timeline
250 AD - 300 AD

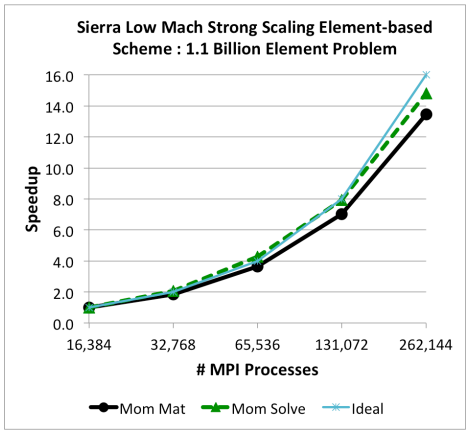
250 AD: The Goths invade Anatolia (AKA, Asia Minor), in sea-borne raids they plunder *Nikopolis*

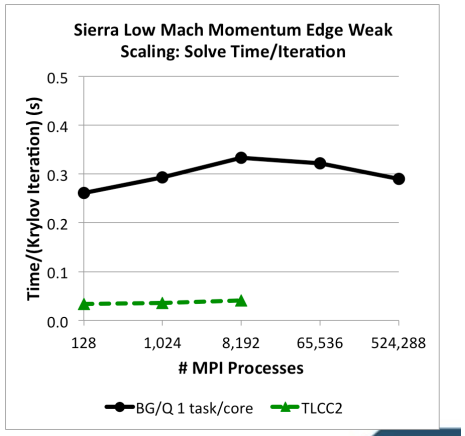
- Trinity open-science Domino and Barone, SAND2016-4085C (2016) to elucidate stall cells
- O(6) billion element wind energy application (high-Re flow past thick blade in high AoA
- O(128,0000) core run for ~one week ~1800 year simulation using computing resources available 10-years ago: Complete in roughly 29 years!

<u>Recent impinging jet study</u>: 200,000,000 CPU hours (~23,000 years)

- 25C, 100C, 300C, and 600C case run on 2000-4000 nodes (32 and 64 mpi ranks/node)
- GPU-based approaches now emerging

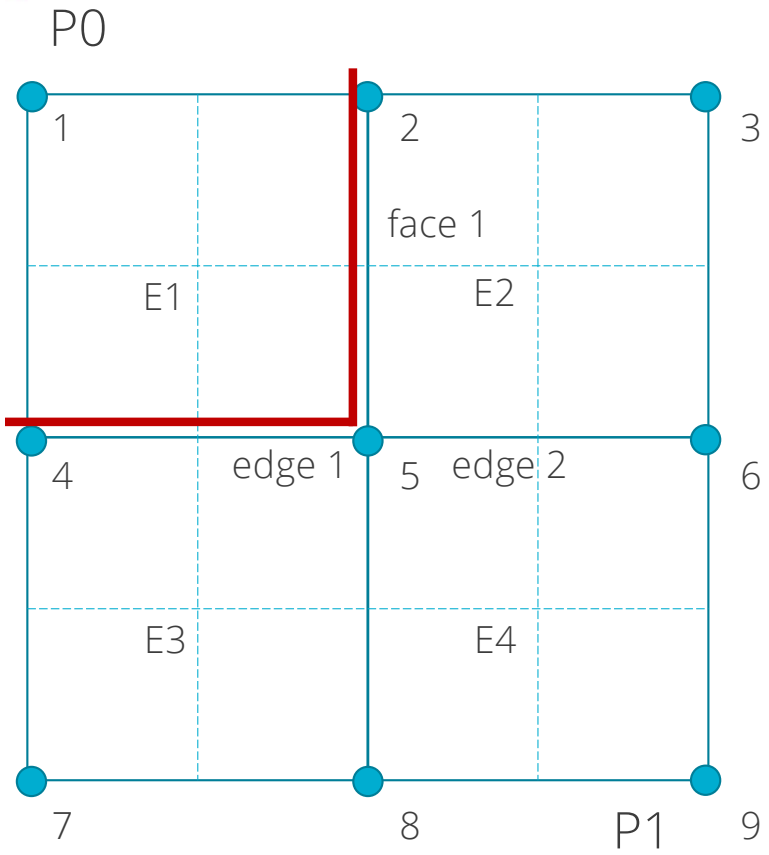
A giant at redwood tree illustrating the passage of time within its' growth rings





Lin et al., Parallel Proc. Letters, (2014)

Parallel Strategies: Option 1



Sierra Toolkit/Standard Ownership rules: Mesh object (nodes, edges, faces, elements) owned by lower-rank. For example:

- P0 locally owns E1 and nodes 1, 2, 4, and 5
- P1 locally owns E2, E3, and E4 and nodes 3, 6, 7, 8, and 9
- P0 and P1 share nodes 2, 4, and 5

Desired Control-Volume Finite Element (CVFEM) Operation:

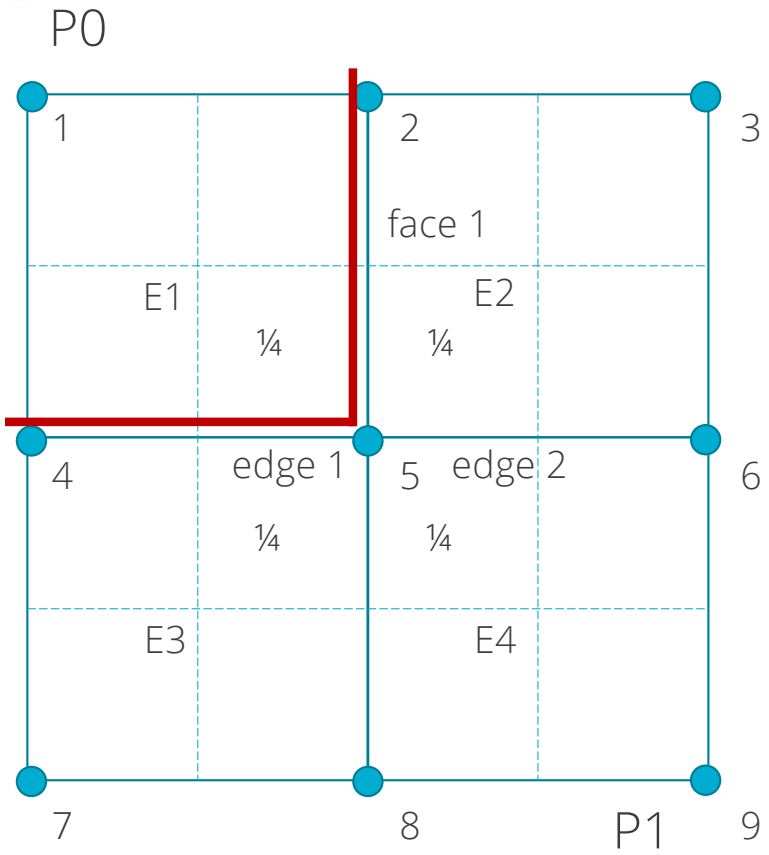
- "nodal volume" assemble for node 5 via iteration of each 1cmx1cm element. Desired nodal volume for node 5 = 1 cm^2 exactly the same on both parallel ranks

Options:

1. Iterate over locally owned elements and locally accumulate nodal volume field followed by a parallel sum and copy owned to shared.

2. Ghost element E2, E3, and E4 to P0 and E1 to P1 (along with coordinates) and iterate locally owned and ghosted elements. Locally accumulate nodal volume field followed by a copy owned to shared.

Parallel Strategies: Option 1



Sierra Toolkit/Standard Ownership rules: Mesh object (nodes, edges, faces, elements) owned by lower-rank. For example:

- P0 locally owns E1 and nodes 1, 2, 4, and 5
- P1 locally owns E2, E3, and E4 and nodes 3, 6, 7, 8, and 9
- P0 and P1 share nodes 2, 4, and 5

Desired Control-Volume Finite Element (CVFEM) Operation:

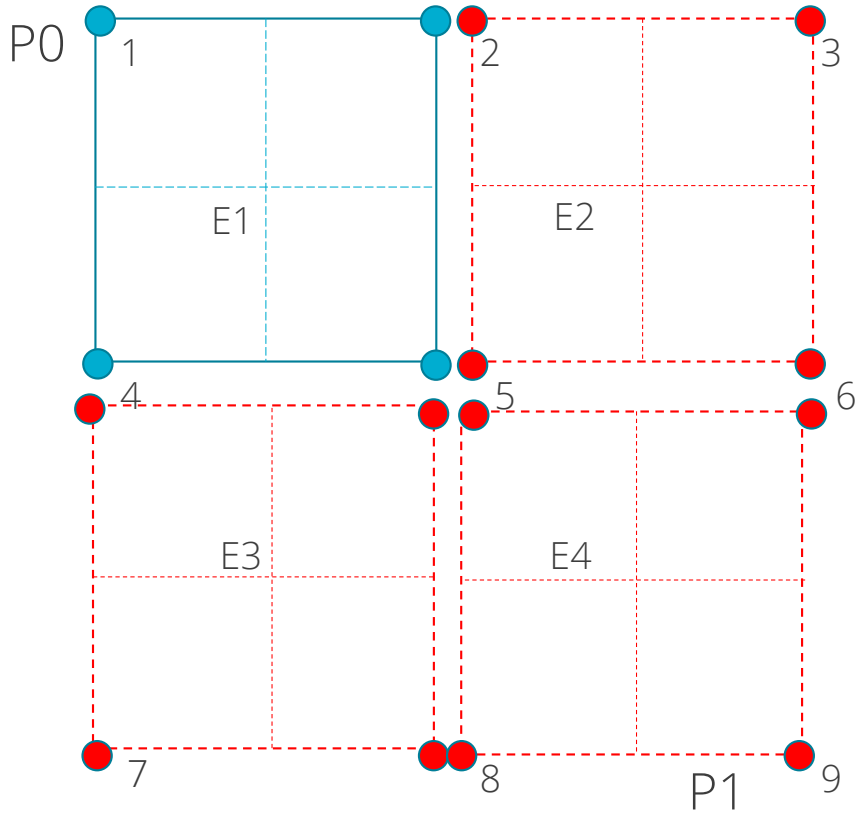
- "nodal volume" assemble for node 5 via iteration of each 1cmx1cm element. Desired nodal volume for node 5 = 1 cm^2 exactly the same on both parallel ranks

Options:

1. Iterate over locally owned elements and locally accumulate nodal volume field followed by a parallel sum and copy owned to shared.

2. Ghost element E2, E3, and E4 to P0 and E1 to P1 (along with coordinates) and iterate locally owned and ghosted elements. Locally accumulate nodal volume field followed by a copy owned to shared.

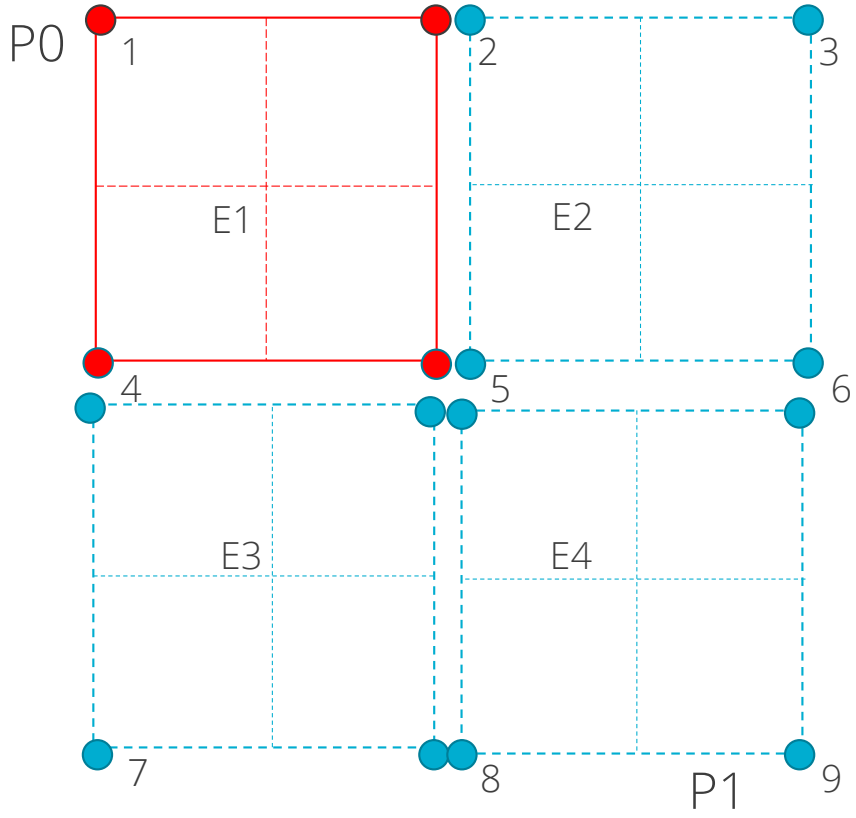
Parallel Strategies: Option 2, A View from P0



Ghost P1 -> P0

- All work performed on P0
- Ghosting and communication required
- No parallel sum (think about a matrix entry)

Parallel Strategies: Option 2, A View from P1

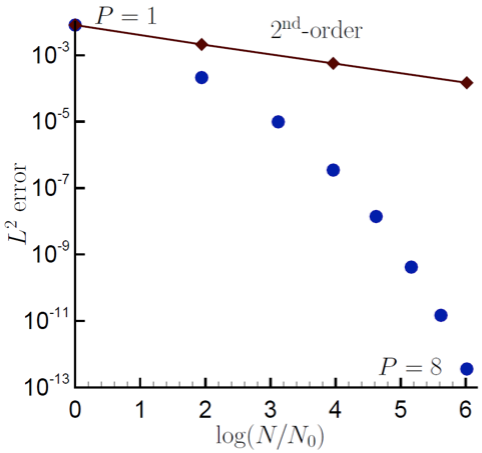


Ghost P0 -> P1

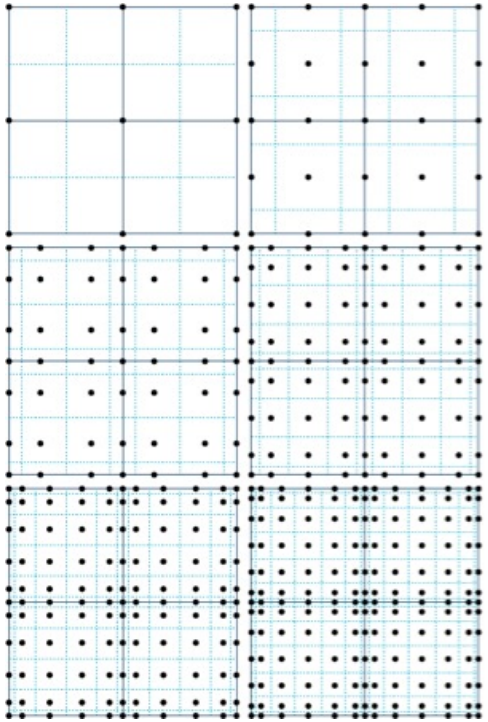
- All work performed on P1
- Ghosting and communication required
- No parallel sum (think about a matrix entry)

Conceptual Disparity Between Low- and High-Order

- Recall, that for unstructured schemes, increased connectivity to achieve a higher-order basis is difficult
- However, we have discussed a notion of "polynomial promotion" of the underlying basis that provides higher-order convergence



Spectral convergence



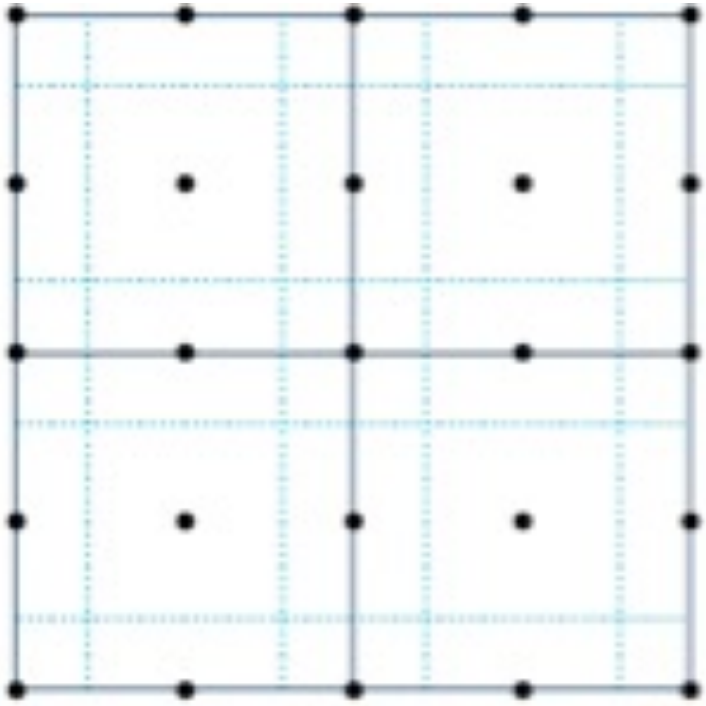
Dual-volume for promoted quad4

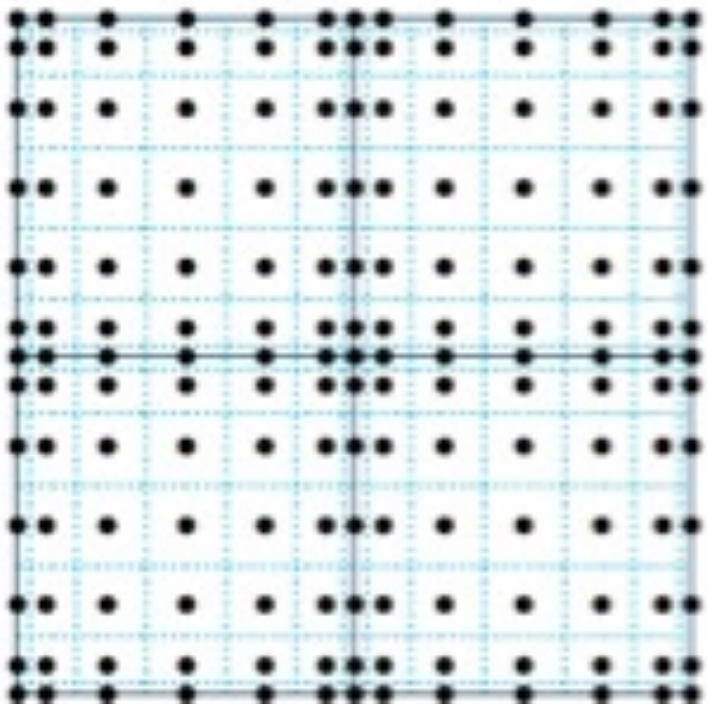
Work!

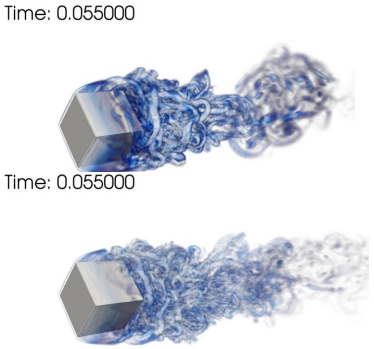
Obviously, the promoted element has far more nodes

Gauss-Lobatto: nodes

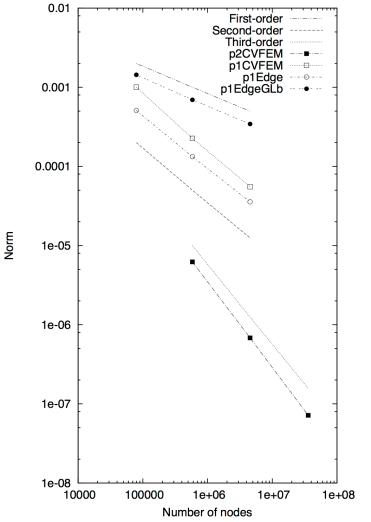
Gauss-Legendre: sub-control surfaces







Rotating cube (Re 4000, RPM 3600) P=1 (top) and P=2 (bottom)



Timely Publications on Next Generation Platforms

Knaus, Robert. 2022. “A Fast Matrix-Free Approach to the High-Order Control Volume Finite Element Method with Application to Low-Mach Flow.” Computers & Fluids, https://doi.org/10.1016/j.compfluid.2022.105408

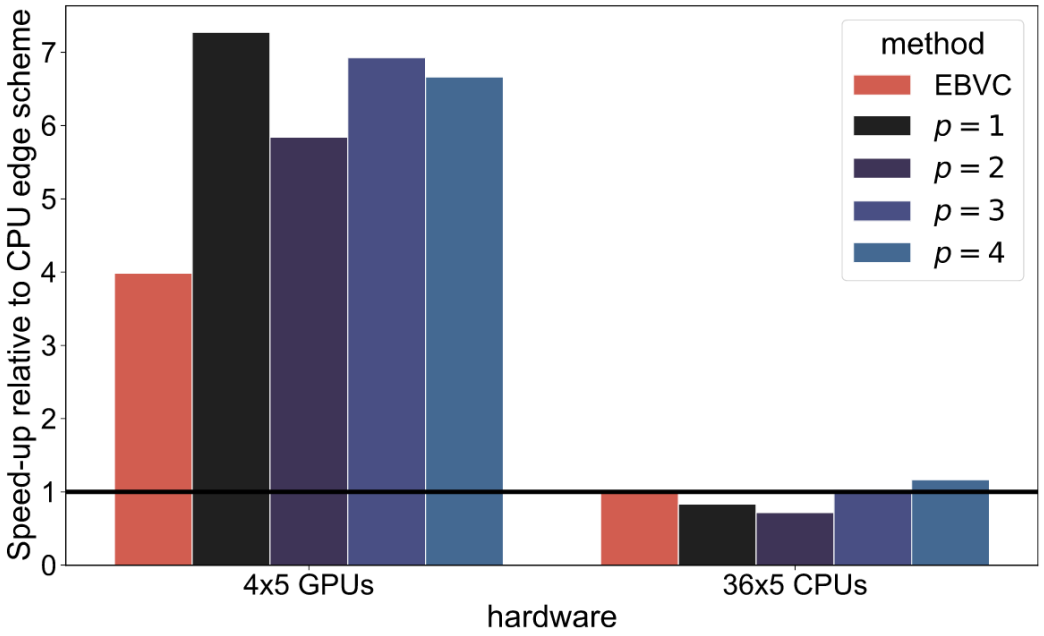


Fig. 10. Wall time per turnover time speed-up, relative to the node-centered finite volume CPU, of different polynomial orders on 5 computational nodes of a system having 4 Nvidia Tesla V100 GPUs per node vs 5 nodes of a system having 36 Intel Broadwell CPU cores per node. For a 50-million point turbulent channel flow LES, the matrix-free method is outperforming the CRS-matrix based node-centered finite volume scheme (EBVC) on the GPU system with similar speeds on the CPU-based system.

High Performance Computing for CFD: Conclusions

- Several fluids applications found in the low-Mach application space require HPC
- Strong and weak scaling are of interest in engineering analysis
- Communication bottlenecks can affect scaling
- From a user-perspective, scaling is critical to efficiently deploying production simulation results
- Path towards NGP will require disruptive technology with significant code investments